Applicants	Lalonde et al.	
Serial No.	10/792,157	**INVENTOR DECLARATION UNDER 37 C.F.R. § 1.131**
Filing Date	3/3/2004	
Confirmation No.	3709	
Examiner Name	Ben C. Wang	
Group Art Unit	2192	
Attorney Docket No.	143.008US01	

Title: A DISTRIBUTED SOFTWARE FABRICATION SYSTEM AND PROCESS FOR FABRICATING BUSINESS APPLICATIONS

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

   We, Jean-Louis Lalonde and Alexandre Boudreault-Ferland, hereby declare the following:

   1.   We are the co-inventors of the subject matter claimed in the above-identified patent application Serial No. 10/792,157 (hereafter "the '157 Application") filed on March 3, 2004.

   2.   Prior to November 2003, we conceived the idea in Canada of a distributed software fabrication system and process for fabricating business applications, as described and claimed in the '157 application.

   3.   Copies of three white papers describing and showing the inventive concepts claimed in the '157 invention and their reduction to practice prior to November 2003 are attached hereto as Exhibits A, B and C.

   4.   The paper of Exhibit A is entitled *"Upgrade Secluded Desktop Applications to XInternet e-Business Applications"* and was published prior to November 2003.

   5.   The paper of Exhibit B is entitled *"XI-Factory^{TM} E-Business Model Design"* and was published prior to November 2003.

6. The paper of Exhibit C is entitled *"A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment"* and was published prior to November 2003.

7. We declare that all statements made herein of our own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful, false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful, false statements may jeopardize the validity of the application or any patent issuing thereon.

Dated this _28_ day of _January_ 2008.

_____

Jean-Louis-LALONDE

Dated this _31_ day of _January_ 2008.

_____

Alexandre Boudreault-Ferland

Achieving Ei and B2Bi, the MDA™ Way!

Upgrade Secluded Desktop Applications to XInternet e-Business Applications

Moving Towards an N-Tier Client-Server Application Consuming Web Services.

JL Lalonde
President, CEO

An AZUR Technologies White Paper

A Z U R
TECHNOLOGIES
www.xiazur.com

Upgrade Secluded Desktop Applications to XInternet e-Business Applications

Table of content

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

Trademarks

The following are trademarks or registered trademarks of their respective companies:

- XI-Factory is a trademark of AZUR Technologies
- XI-Framework is a trademark of AZUR Technologies
- OMG is a trademark of the Object Management Group
- MDA is a trademark of the Object Management Group
- UML is a trademark of the Object Management Group
- Visual Studio is a trademark of Microsoft
- Visual Basic is a trademark of Microsoft
- Visual C# is a trademark of Microsoft
- Microsoft Access is a trademark of Microsoft

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

Abstract

Enterprise integration (Ei) and Business to Business integration (B2Bi) are best achieved when business needs are driving the IT activities. Currently, one important IT activity is to upgrade obsolete desktop applications to the latest version of web-enabled technologies.

Desktop applications are simply programs that are installed and run on your PC. Many of your old desktop applications are built from a variety of generic technologies (such as spreadsheets and database management systems). The industry is currently overflowing with millions of such legacy applications that are reaching the end of their life cycle.

To stay current with ever-evolving business needs, these applications must be quickly recycled to a web-based format, but at a fraction of the previous development cost. Moreover, many corporations are recognizing the need to renew legacy mainframe applications by providing a web interface and to improve interconnections with clients and partners.

> "As applications get older, their speed and power decreases, their instability increases, and the maintenance costs continue to soar"

Today's users are all too aware of the limitations that shackle their desktop applications, keeping them from providing the much-needed solutions to their business needs. As these applications get older, their speed and power decreases, their instability increases, and the maintenance costs continue to soar, especially when they apply to multiple users across a rising number of workstations. As the number of users continues to grow and as they become more and more efficient at their jobs, they require more and more real-time access to critical business information from the enterprise intranet or via the Internet.

On top of that, the installed technology continues to lag behind needs of its business users. Most business users complain that the level of business integration the applications can provide is not acceptable.

Out with the Secluded, In with the Distributed

In the past, the only way to take advantage of the whole new world of technology that was out there would be to write your own application from scratch. Not only is this approach highly expensive and time consuming, but you would run the high risk of not being able to upgrade your application and have it be compatible with the latest set of technologies.

Today, you can use modern conversion tools to upgrade the power of your old applications to be at the same level as the latest set of technology. However, the

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

problem with these tools is that they operate at the code level and were not designed to raise these applications to the web service/N-tier client-server model level, which is better suited to support dynamic e-business applications.

Techniques to Upgrade Economically

This White Paper discusses the fundamentals of how to economically upgrade desktop applications to a web-compatible format using standard conversion tools enhanced by a Model Driven Architecture™ approach. This White Paper will present the new techniques that can minimize the distortion of the application's appearance in its new web projection and that can better interconnect distributed business processes.

It will also illustrate how the AZUR Solution can support the upgrade processes to reduce complexity, time, and cost, while still promoting the business strategic alignment.

Finally, to better appreciate the upgrade process, this White Paper will explain how to upgrade, in only a few hours, the well-known Microsoft Access Northwind client-server application to a full XInternet e-business application working with the latest Microsoft .NET technologies.

The Upgrading Challenge

Before describing a new method to upgrade a desktop application to the web, you need to understand the challenges involved. Dynamic client-server desktop applications are not easy to upgrade and to deploy over the web. They require a much better software and deployment infrastructure than the regular informational web site that only displays static web pages.

While most desktop applications upgrades target the Enterprise intranet (Ei), some are targeted solely to the Internet (B2Bi). Moreover, some will be deployed on both types of networks, for example: Order Processing at the Intranet level and Order Tracking offered to clients over the Internet.

The Limitations of Common Web Architectures

There are several types of web architectures to which a typical dynamic desktop application can be converted. One of the more common types is a web browser accessing dynamic web pages (using PHP or Active Server Page (ASP)) while connecting to the database frequently with SQL embedded into the page. However, most companies have found that this method is no longer acceptable because of the obvious security breaches. Moreover, the application development, debugging, and

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

maintenance problems associated with hundreds of dynamic web pages are beyond the capabilities of medium to complex business applications development.

Most dynamic web applications have been built on this type of web architecture despite important technical limitations. Although this approach is suitable for low security, general information presentations and small applications, it is not appropriate for modern dynamic e-business applications consuming web services that require high performance and security levels while accessing enterprise databases in real-time.

Some of the limitations of the regular dynamic web page model are as follows:

- Low security when not isolated from the database.
- Highly instability when multifaceted web applications require many pages with complex workflows.
- Difficult to code, debug, maintain, and support.
- Difficult to reproduce a desktop application's workflow, its look-and-feel, and its performance.
- Business rules are coded directly into the script files.
- If poorly designed, the SQL code is located in the script files.

Another important problem is related to the secluded nature of static web sites. Currently, most web sites are not interconnected at the business level and while desktop applications can be upgraded through the web, there are still largely isolated. Although extended networks have increased their speed and capacity, web applications still lack the ability to properly interrelate with each other.

Clearly, to succeed in the field of Ei and B2Bi, we need to improve the web infrastructure and eliminate such limitations. In the following sections, we propose a modern approach to the application upgrade process and answer the following questions:

- Why should you upgrade from a desktop application to a web-based application?
- Which application elements must be upgraded?
- What is the modern way to upgrade the application?

Upgrading from the Desktop to the Web

At the center of an enterprise's preoccupations is the renewal of legacy mainframe and desktop applications while maintaining IT and business units' strategic alignment. IT organizations worldwide use a mix of heterogeneous operating systems (OS/390, UNIX, Windows) to run their most complex applications. The need to protect and extend these systems, while tapping new technology, is central to their future successes.

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

But you still need to be able to make clear case to justify why you should convert your desktop applications to the web. Not all applications need to be converted and some very specific applications that are only used by a few employees are not good candidates.

On the other hand, there are two important types of reasons to convert old and secluded applications to the web:

- **Business**: Pools of users are requesting it since they know that Ei and B2Bi can be best achieved through distributed business processes and securely reachable from the intranet or the Internet.

- **Technology**: Business strategies are best met using modern IT technology strategies, which in turn are highly supportive of business needs.

  More than ever before, IT departments are using web-enabled technologies to ensure business processes can interoperate quickly to reach the highest level of Enterprise integration and at the lowest cost possible.

Another trend comes from the fact that desktop applications have been known to inherently grow and become more complex over time. A small application that started as a simple, low scale prototype can grow to become a critical business application supporting tens and sometime hundreds of users!

Even now, hundreds of thousands of enterprises currently have several of these critical applications running on remote client-server platforms. What they desperately need to remain efficient is to scale themselves to the web so that their applications can grow larger and more complex, while still supporting a rising numbers of distributed users.

A typical desktop application runs onto a two-tier client-server with a close tie to the physical implementation. A desktop machine working as a client is physically combined with a network server hosting the business database.

In a traditional client-server model, business rules, or application intelligence, are split between these two complementary poles. Most of the time, around 70% of the business rules reside at the client side frequently refereed as a Fat or Intelligent Client. The client side can be coded in Visual Basic or other similar 4GL. The server side holds the DBMS with contains stored procedures almost always directly bounded to user interface controls.

Desktop applications that are integrated into a similar client-server topology have been plagued with the following limitations:

1. Difficult to deploy on every workstation and come with very high support and maintenance costs.

2. Not scalable because the application cannot grow beyond the physical boundaries of a typical two tiers client-server platform.

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

3. Not reusable since business rules cannot be encapsulated and centralized onto a single business server.

4. Poor performance over time because the application grows and becomes more complex, consuming more and more of the limited client workstation processing power and resources.

With all of these factors in mind, you can clearly make a solid justified case to proceed with the application conversion. But before making that move, you must decide which parts of the application need to be upgraded?

Deciding which Application Elements to Upgrade

To better assess the scope of a typical application upgrade to the web, there are several application layers, elements, and components that need to be translated to the newer set of web enabled technologies.

The following is a list of typical **software elements** to be enhanced, converted, migrated, and aligned to business needs:

1. **Software basic infrastructure conversions:** from isolated and non-standard technologies to web-enabled technologies supported by standard protocols, data structures, etc.

2. **Applicative infrastructure conversions:** from a two-tier client-server to N-tier client-server topology supporting a high level of reuse through web services.

3. **Application presentation conversions:** from a richer client to a somewhat thinner web-based presentation build using secure technologies like HTML, JavaScript, and XML.

4. Application object model, logic, workflow, and business rules conversions and adaptations.

5. **Security scheme conversions:**
   • Improve the security model.
   • Replace the database security scheme with a distributed security web service and component managing user identification, authorization, and roles.

6. Report conversions.

7. System integration (ERP, CRM, etc.) migrations.

8. May require some database relational model and table, triggers, transactions and stored procedures adaptations and migrations.

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

9. May require some enhancement to the application to promote business strategic alignment.

And this is just a short list: there are definitely more conversions and adaptations to make. With so many elements to upgrade, you cannot start from scratch with such a huge project. Chances are the application was built and customized over a period of many years. But how can you regenerate this application in only a few months at a reasonable cost?

In the next section, we present an MDA approach from which we can define the new application from its old business model signature and regenerate it into a modern web infrastructure.

Depending on the size of the application, this can be achieved quickly providing you have the proper software process and tools to support it. After this initial phase, you can apply standard conversion tools to the best candidate elements to complete the work.

The Best Way to Upgrade an Application

Once you get a feel of what needs to be upgraded, you must evaluate the methods to transform an older application into a more distributed web technology format.

There are various methods and techniques that can be used to renew the application, but in many cases, these approaches can be combined. A redesign may be better for some structural elements whereas many others might need only language syntax level conversion.

There are two main strategies to upgrade client-server application to the web. You can use a **horizontal** and/or a **vertical** upgrade.

- **Horizontal conversion**: one layer of the application is converted into a matching web technologies followed by other layers.

  For example, you can migrate the user interface to HTML and JavaScript code, while the middle tier would be converted in VB.NET or C# code (which can be accessed through web services).

  Horizontal migration is ideal in situations where the modules are tightly coupled and the effort and risk involved in the migration is known to a large extent.

- **Vertical conversion**: all the tiers of a software component of a standalone application are migrated to the most appropriate web technologies.

  Vertical migration can be adopted in situations where the components are mostly loosely coupled or the risks and effort involved in a total migration are unclear. By migrating one module, you can get a good idea of the effort required to migrate other modules.

9 of 20

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

For example, Visual Studio.NET provides a Visual Basic Upgrade Wizard which performs almost the entire vertical language conversion, barring some modifications that must be made to complete and optimize the upgrade process.

Actually both these strategies can help regenerate the application from a strategic and tactical technique point-of-view.

Redesigning the applicative infrastructure is required if you need to use the new features of the .NET architectural framework (system level). Managed code can easily run and interoperate with unmanaged code (older COM components). Therefore it is possible to reuse many existing components as such.

However, paradigm shifts towards XML web services, Web Forms, and the One Web Page Application call for some horizontal rework in design, coding, and deployment. Standard application conversion tools have inherent limits because they start from the code level instead of the model level.

In the following sections, we present several modern techniques in order to maximize the application upgrade through horizontal and vertical upgrade processes.

Model Driven Architecture™

Model Driven Architecture™ (MDA) has been developed by the Object Management Group (see: www.omg.org) to maximize the isolation of business models and requirements from the ever-evolving enterprise underlying information technologies. MDA promotes XML Web services that are based on traceable business requirements, as well as promotes the automatic application code generation onto specific technologies at a given time. MDA eases the upgrade of business applications onto a new set of technologies while promoting the strategic alignment of information systems strategy with business strategy.

An MDA approach is a highly horizontal upgrade process. It can be used to quickly regenerate the application and profit from a new set of technologies. A well-defined MDA upgrade process, with tools, can convert most of the software elements as described in a previous section of this White paper.

Thus, with a MDA bridge you can upgrade your application client-server platform and technology based on the latest application business models. As technologies evolve rapidly, the enterprise business models are less subject to drastic and profound changes.

This is the foundation of the MDA approach. Its principle is simple: It develops your business application models in terms of your company objectives rather than the technical environment. It consists in creating a PIM (Platform Independent Model) for the company as well as one or more PSMs (Platform Specific Models) in terms of the

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

company's technical environment. The language of model definition is UML (Unified Modeling Language), which was adopted as the standard by OMG in 1997.

XInternet

The latest trend that is gaining increasing popularity is the move from the publishing of static pages to Internet networks towards the more dynamic e-business XInternet ('eXecutable' Internet, see www.forrester.com). The Internet (as we know it today) has several limitations that prohibit it to efficiently achieve Ei and B2Bi. Here are some limitations that need to be resolved in order to efficiently upgrade a desktop application to the web.

To Avoid Managing too many Web Pages

One critical application layer to upgrade to the web is the application user interface with its underlying software infrastructure. Over the years, people have been somewhat accustomed to the web limitations. They have found a comfort zone for static information publishing. However, for dynamic desktop application upgrade to the web, the transition is not as smooth. This is because desktop applications, accessing corporate databases, are much more complexes than regular informational sites. Up to now, web technologies have not adequately matched the level of desktop application natural complexity.

With hundreds of web pages working in conjunction to replicate the client environment, it simply cannot handle complex application workflows efficiently. This approach has proven itself to be highly unstable, and although there are several reductive techniques that can minimize the conflicts, there are still disastrous side-effects. For example, complex workflows are often reduced in complexity by forcing users to follow a simpler but somewhat cumbersome irritating web navigation path. This is to avoid managing all possible user generated states and conditions at the client side.

As a result, when a condition occurs that the application cannot handle, the user is forced to restart the web session from the initial entry point.

The following is a list of some additional limitations associated to standard web architecture compared to its desktop counterpart:

- Web client controls are not as rich
- The application's look-and-feel is distorted and deformed
- Performance and robustness is noticeably lacking
- Reliable security is more difficult to achieve

To improve Interconnectivity at the Business Level

Another important problem is related to the secluded nature of static web sites. Currently, most web sites are not interconnected at the business level. Desktop applications upgraded to the web are almost as isolated as they were when they

11 of 20

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

existed solely on the workstation. Although extended networks have improved in speed, web applications still lack the ability to properly interrelate with each other.

But thanks to the new paradigm shift toward the XInternet, web applications can regain the same power and performance as regular desktop applications. Moreover, with Web services at the central part, it enables cheaper, shareable, and flexible links to customers and partners.

The XInternet is currently replacing today's static web pages with more dynamic techniques, as well as replacing the fragmented IT infrastructure inside companies with interconnected processes over extended networks.

Web Services

Web services provide a standard means of interoperating between different software applications running on a variety of platforms. Web Services use SOAP (Simple Object Access Protocol see www.w3.org/TR/SOAP) to communicate with XML-based messages to achieve dynamic integration between two applications. Web Services have been designed to promote the dynamic Ei and B2Bi.

Several enterprises are currently integrating Web Services into their Enterprise Integration/Business to Business Integration (EI/B2Bi) strategy. Web Services technology can add value in the following ways:

- Web Services have a low barrier to entry for development teams.
- Web Services are independent of operating systems and language.
- Web Services are based on standards that will likely be implemented by most companies to support their supply chain.

N-Tier Client-Server Model

The limited effectiveness of two-tier client-server model in a highly distributed environment has brought to us the new and improved N-tier client-server model. This enhanced client-server model is based on the ability to construct partitioned applications which in turn can be easily reachable form the web.

Partitioning an application breaks up your code into logical components disseminated on business and web servers. Components can be logically grouped into three tiers:

- User services
- Business services
- Data services

Once an application has been constructed using this model and its supporting applicative infrastructure, each component can be deployed to any machine which will provide the highest level scalability, security, robustness, and performance. There are essentially four benefits to N-tier client-server architecture model:

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

- **Performance:** Given time, any desktop application will outgrow its desktop machine. But thanks to N-tier client/server model, you can deploy your application components to more than just client workstations.

  You have the ability to shift processing load from a client machine (that may be underpowered) to a server with extra computer processing power and memory, thus significantly enhancing the user experience without any application code optimization.

- **Reuse:** Since business and system components can be centralized onto a few servers, any application being built can reuse their distributed services. You build a component once and reuse it constantly in current and future development.

- **Manageability:** Large software programs (.exe) are divided into a set of more manageable components.

- **Maintenance:** Centralized components are much easier to upgrade and deploy when a modification is made. Also, a browser application costs much less to maintain.

Before we can upgrade a regular client-server application to the web, there are several important design issues we need to take into consideration. Ideally, we would design an application with an N-tier client/server infrastructure in mind from the very beginning. The design to build a dynamic e-business application is often very different than a file server-only application. If you have an existing application that was not built with N-tier client/server in mind, you will likely need to retrofit your application in order to take advantage of the upsizing process to follow.

This is exactly how a MDA™ approach can quickly convert a regular client-server application to a web-enabled N-tier client-server applicative infrastructure.

.NET Framework

The question that immediately leaps to mind is "Why do I need to convert what I have to the .NET technology?" The power of the .NET architecture lies in its Managed Code feature, which adds to the robustness of Java applications, especially when you compare them to the previous generations of Microsoft software. VB.NET code targets the Common Language Runtime (CLR) by compiling into an intermediate language, which is then executed under strict control (managed). CLR manages the code in a very similar fashion to the Java model, thereby making applications much more Robust, Stable and secure. The application also becomes more maintainable because of the managed code.

XML Web Services creation and integration as facilitated by the .NET platform are readily available through VB.NET. This makes the creation of hosted applications easier, thereby providing wider access to proprietary intellectual property. XML and web services also enable Enterprise Application Integration through standard methodologies as available through the Microsoft family of .NET servers.

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

Conversion tools

In general when using the MDA approach with tools, you can expect to generate 40% to 75% of the application code from the business models and requirements. Specific structures like business rules can not be easily generated automatically. You always need to use a Software Development Process to manually finish the work.

In order to quickly convert the entire application to the web, you obviously need a modern development environment combined with some upgrade wizard. The next section presents the AZUR Solution as modern upgrade process.

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

The AZUR Solution

The AZUR Solution aims to simplify building e-business applications and facilitate integrating your company's business processes by using XML Web services. The AZUR Solution is composed of three main vectors:

- **Software Fabrication Process (SFP) with productivity tools**: SFP speeds up and supports the automated fabrication of e-business applications while reducing both the time and cost required to complete these projects.
  - For more information, see the AZUR White Paper "A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment".
- **XI-Factory™**: An MDA-driven web application that can quickly define, validate, and automatically generate operational .NET solutions from business models.
- **XI-Framework™**: A complete N-Tier applicative infrastructure that supports XInternet e-business applications generated on .NET technologies.

The AZUR Solution offers web services, in an ASP (Application Service Providers) mode, and provides the following free business benefits to the business and IT community:

- Share and define public business models
- Rapid prototyping of e-business applications
- Rapid user validation of e-business applications
- Go-No-Go type of web tools

Once a project is approved and purchased, the client can automatically generate and download his operational .NET solution to be completed into his own development environment.

The types of dynamic e-business applications that the **AZUR Solution** can produce and upgrade are as flexible as your needs:

- **Lightweight application**: responding to the needs of a few people with very short project cycles. For example, you can upgrade a Microsoft Access application to XInternet e-business application.
- **Highly Complex application**: responding to the broader needs of larger projects such as e-order tracking, e-CRM, or financial applications that manage billions of dollars.

In addition, to achieve XIntenet benefits for our clients, we have defined the **One Web Page Application** (OWPA) concept and constructed its infrastructure on top of web services. As it names implies, the OWPA concept only uses one ASPx file on the web server to generate the user interface into the Internet Explorer web browser. This helps

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

to greatly reduce the development complexity while scientifically enhancing the application performance.

Another important advantage of the OWPA is related to its design: It replicates the power, look-and-feel, and ease of use of a regular desktop application, but deploys it over the web. It also tries to reduce, as much as possible, the distortion between the old client application projections against the new web application projections.

The OWPA concept offers e-business benefits that supply the **same power and performance as your regular desktop applications.** In the time of XML Web services, we perfected this innovative concept to offer you a richer Web projection that is better adapted to the natural complexity of your company's business functions.

The OWPA concept derives its originality from two sources:

- the single Web page that is built once at the beginning of the process
- the bi-directional communication between the client's workstation and the Web server

This communication rides on a SOAP protocol and uses Web services that transfer small XML documents related to the application's data and actions, ensuring fast execution speed. This innovative Web approach is different from anything else you have come to know about regular Web applications. **An application based on this concept requires no page transition when you perform a request!**

In the following section, we show how to upgrade the well-known Microsoft Access Northwind application by transforming it into the **XI-Northwind** e-Business Application.

Upgrading the Microsoft Access Northwind Application

The purpose of this section is to demonstrate how a Microsoft Access-based application can be quickly upgraded to the web as a full XInternet e-business application.

Although Microsoft Access is powerful enough in its current form, as Microsoft's .Net grows more popular, so does the demand to upgrade it's isolated yet critical Access applications to more accessible and better integrated solutions.

Microsoft Access is a powerful file server database management system that, in addition to allowing a developer to quickly build low-scale solutions, also offers the ability to rapidly prototype client/server solutions for prospective clients. However, in a modern Ei and B2Bi context, a file server database or even a conventional client/server database supporting critical business applications is not powerful enough.

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

Microsoft Access Limitations

Generally access and performance problems with MS access comes with multiple users (> 5) accessing the database at the same time. When that happens, Access' limitations can cause the following problems:

- If the database grows beyond its limit or beyond the available disk space, then database itself can be corrupted, even to the point of being un-recoverable. Access' security mechanism is not very robust.
- There are no backup mechanisms for MS Access, nor are there transaction logs.

    As a result, the technology cannot guarantee the integrity of any backup and restoring upgrades of Access can be difficult.

In this regard, it is quite obvious why business critical Access applications should be upgraded to a better web application infrastructure.

The Upgrade Process

With the AZUR XI-Factory web application, we applied a horizontal upgrade on the MS Access Northwind application. Using the XI-Factory as an MDA .NET Bridge provides the ability to rapidly prototype XInternet e-business application consuming web services built on N-Tier Client/Server infrastructures that allow for better Ei and B2Bi opportunities.

- The first step was to define a new solution called XI-Northwind.

- The second step was to construct the business model within the XI-Factory Constructor.

- The final step was to generate the .NET solution.

Therefore, we upgraded the user interface to HTML and JavaScript code, while the middle tier was converted in VB.NET code accessed through web services. The database tables and stored procedures have been generated using the MS SQLServer DBMS.

Since the Northwind application does not contain many business rules, we used Visual Studio .NET to apply a vertical conversion on one report to get an accurate idea of the effort required to convert other reports.

The Upgrade Results

The following figure shows the final results obtained immediately after the application generation. We added some data to better visualize the seven resulting XI-Northwind e-business applications:

1. Customer Management

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

2. Product Management

3. Order Management

4. Supplier Management

5. Shipper Management

6. Employee Management

7. Product Category Management

As you can see from this screen snap, we generated seven One Web Page Applications (OWPA) within the same web session so that the user can access them easily.
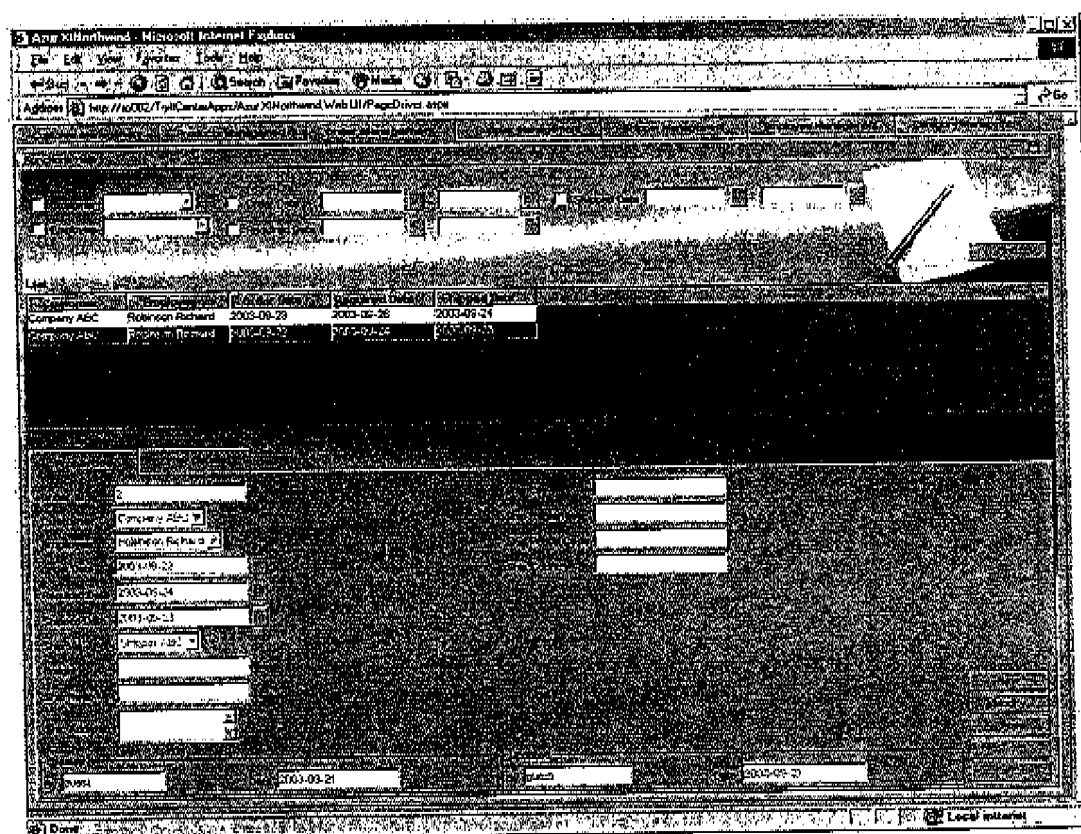
Another important advantage of the OWPA is related to its design: it replicates the power, look-and-feel, and ease-of-use of a traditional desktop application, but deploys it over the web. It also reduces, as much as possible, the distortion between the old client application projections against the new web application projections.

The OWPA concept offers e-business benefits that supply the same power and performance as your regular traditional desktop applications.

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

The following table illustrates the upgrade phases with their associated activities and time metrics.

Phases	Description	Time required
Specification	Install and understand MS Access Northwind application	30 minutes
Analysis	Analyse the application	30 minutes
	Construct the application	2 hours
	Validate the application	30 minutes
	Generate the seven e-business application code and import it into the Visual Studio developer environment.	5 minutes
	Compile the application.	
Design	One Report	15 minutes
Code	No additional code required.	
	Coded a report	1 hour
Test		30 minutes
Deployment		30 minutes
Total		5 h and 50 minutes

Conclusion

This White Paper has presented the fundamentals of how to economically upgrade desktop applications to a web-compatible format using standard conversion tools enhanced by a Model Driven Architecture™ approach. We have presented the One Web Page Application (OWPA) as a new XInternet technique that can minimize the distortion of the application's appearance in its new web projection and that can better interconnect the distributed business processes through web services.

We have shown how the AZUR Solution used as an MDA .NET bridge can support the upgrade processes to reduce complexity, time, and cost, while still promoting the business strategic alignment.

Finally, to better appreciate the upgrade process, this White Paper has shown how, in the space of only a few hours, we were able to upgrade the well-known Microsoft Access Northwind client-server application to a full XInternet e-business application working with the latest Microsoft .NET technologies.

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

You are invited to register yourself and try by yourself the AZUR XI-Factory power at our web site: www.xiazur.com. You can also test several Dynamic e-Businesses Sample Applications to better evaluate the XInternet sensation and performance at the AZUR Application TRY IT CENTER.

Achieving Ei and B2Bi, the MDA™ Way!

XI-Factory™ E-Business Model Design

Expanding Upon UML, Entity-Relationship and Relational Models

JL Lalonde
President, CEO

An AZUR Technologies White paper

AZUR
TECHNOLOGIES
www.xiazur.com

Trademarks

The following are trademarks or registered trademarks of their respective companies:

- XI-Factory is a trademark of AZUR Technologies
- XI-Framework is a trademark of AZUR Technologies
- OMG is a trademark of the Object Management Group
- MDA is a trademark of the Object Management Group
- UML is a trademark of the Object Management Group

Table of Contents

Abstract

In the last 10 years, the evolution of the Internet has forced software to become increasingly complex to be able to harness the web's untapped potential. The process of building and deploying complex systems has made a quantum-leap forward from object-oriented techniques to visual modeling and process automation. However, these innovations have been mostly centered on the IT side of the equation. The business side of the equation is still very open for discussion!

To achieve the IT strategic alignment with Business Units goals, the pivotal bi-directional communication channel now needs to be addressed:

- How can domain expert users be actively involved in ongoing system validation to ensure properly alignment with their business needs?

- How can IT quickly and cost-efficiently deliver accurate user-centric system details to ensure high harmonization with business needs?

To definitively answer these complementary and fundamental questions, we need to create a set of efficient web services that feature a unique e-collaborative environment for Business Users. These services must quickly validate and securely transmit IT business analyst software prototypes, business processes, tasks, and rules assumptions via the web.

Get Involved with XI-Factory

To deliver a solid foundation for improving Business User involvement, we provide the XI-Factory™. A web application that offers an innovative set of web services designed for Ei and B2Bi via a web-based *Software Fabrication Process* (e-SFP), which can be used in any place, using any path, at any pace, and at any time.

What makes the XI-Factory unique is that it empowers IT and Business people with a highly e-collaborative workflow to quickly create, reuse and automatically transform business models into easy-to-understand e-business application prototypes, which Business Users can then rapidly test, validate, rectify, and approve over a secured web site.

At the end of the analysis phase a working and well-formed Visual Studio .Net solution is generated for the developers to download, use and complete within a standard desktop *Software Development Process*.

This paper focuses on AZUR's support for the SFP E-Business Model Design expanding upon UML, Entity-Relationship, and Relational models.

Problem Definition

Since 1995, we have taken on the challenge of E-Business Model Design. For too long, the IT industry has been setting the boundaries around software development for many IT professionals. But the Internet is tearing down many of these boundaries, paving the way for new opportunities for the emerging e-Software Fabrication Process that allows both IT and Business Users to better collaborate on projects over the web.

For most companies, developing software is an art form that needs to move closer to an engineering discipline. Moreover, Business Users are not currently involved enough in the standard Software Development Life Cycle.

The XI-Factory web tool distinguishes itself by catering to the needs of a truly diverse group of people within the IT and Business communities, while providing special attention to the Business Users. Historically, the role of these Business Users has been under-developed due to the restrictive boundaries inherent in the secluded desktop Software Development Process (SDP).

Incorporating All Levels

To successfully define, validate, and generate operational E-business applications from an MDA™ approach, we need to harmonize the functional definitions of the following service levels:

- User Services
- Business Services
- Data Services

The UML notation is well-adapted for Object-Oriented system modeling when performed by qualified IT personnel. UML is designed to accurately depict a system's layout including:

- Class-Object
- Association-Link
- UseCase-UseCaseInstance
- Message-Stimulus
- and so on

The Entity-Relationship (E-R) data model is very useful to map an Object model onto a database Relational model. In a standard Software Development Life Cycle, the application team will commonly use UML to define the application artifacts and the database designers use the E-R and Relational models to define and create the application database.

However, in current Software Engineering–State practice, UML, E-R, and Relational Models cannot efficiently demonstrate the business model requirements to the users. The general consensus is that most Business Users cannot validate a system using these models, underlying notations, and diagrams.

From our previous client projects, we seen it proven that the best way to build a useful system is to present Business Users with a working prototype of the application very early in the Software Development Life Cycle.

The XI-Factory web tool provides a unique bridge between the Object Model and the Business Model. To help you understand the design elements supported by the XI-Factory, we have provided you with the **AZUR.ModelDesignExamples** (AZUR.MDE) solution. Using these standard sample designs, you can learn how to build your own solution E-Business Model Designs. You can also refer to the extensive XI-Factory E-Business model library for other practical examples.

E-Business Model Design

Just like UML ([3], [5]) and the Entity-Relationship Model [2] help to map the object model onto the relational model, we needed an efficient way to map the object model onto the business model. Our XI-Factory web tool has been designed to accurately provide this type of mapping. The ultimate goal is to create and automatically generate e-business application operational prototypes using typical E-Business Model designs.

The typical IT business analyst may not be proficient in UML, so we adapted a design notation that can be compatible with a broad set of typical IT personnel. We have based the XI-Factory on UML, Entity-Relationship, and Relational models to create an efficient MDA bridge to .NET technologies. In the following sections, we introduce the XI-Factory model design elements.

XI-Factory™ Workflow

As described in the AZUR White paper *"A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment"*, the XI-Factory workflow is defined in four straightforward steps:

1. Solution:
   - Define solution parameters
   - Assign user access rights
2. Construction:
   - **E-Business Models**
   - Preview Project Plan Summary

XI-Factory™ E-Business Model Design

3. Validation:
   - XI-Collaboration Center to provide user and analyst feedback
   - **Preview Application Operational Prototypes**
   - Preview Application Functional Documents (in HTML format)

4. Generation
   - **Operational .NET Solutions with Projects, Classes and Databases Artifacts**
   - Application Functional Documents (in MS Word format)

Using this workflow, you can **Construct** your **E-Business Models** and **Validate** them from the resulting **Application Operational Prototypes**. These two highly complementary functions can efficiently help you map the Object Model onto the Business Model, which allows Business Users to validate and customize the final product according to their requirements.

As seen in the following illustration of the **AZUR.ModelDesignExamples** (AZUR.MDE) solution, the **Construct** web page contains the following fundamental hierarchical structural elements:

- A Solution is composed of one or more Applications
- An Application manages one Main (Strong) entity
- A Main entity is composed of Attributes, Actions, and other related entities (optional)
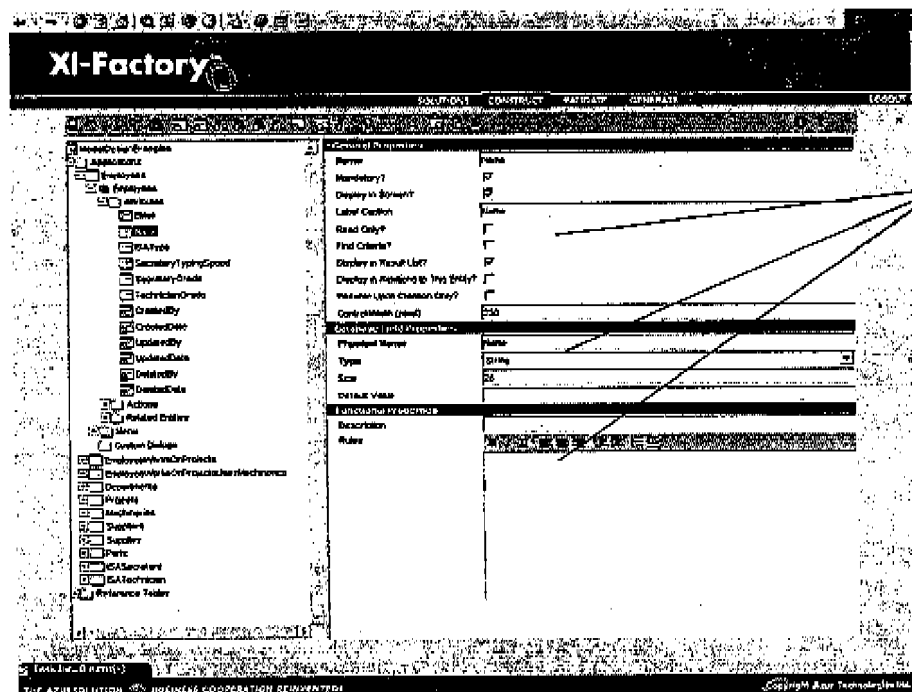
You can customize each item in the tree using the General, Database, and Functional properties.

XI-Factory™ UML, E-R and Relational Models Support

As defined by UML pioneer Grady Booch [7]:

"Object-Oriented Analysis is a method of analysis that examines requirements from the perspective of the classes and objects found in the vocabulary of the problem domain".

Therefore:

- An XI-Factory Construct web page can be seen as a set of web services that help you define and connect **Application Domain Classes (Entities)**.
- The XI-Factory Validate web page can be seen as a set of web services that help you to preview and play with the **Application Domain Objects**.

At the XI-Factory level, the objective is not to fully support the UML or E-R detailed notations, but rather to rely on their pivotal design elements to quickly build, preview, and test business models from the business user's perspective.

From our market studies on Business Model Design activities, we have discovered that most of the IT business analysts and users related better to the Entity concept instead of the more abstract Class concept. Therefore, We have incorporated the Entity concept combined with Actions (Business Operations) into the XI-Factory application constructor.

The XI-Factory also includes elements of the Relational model (Database Properties) to enable automatic application database generation. In the following sections, we explain the underlying model design elements featured in XI-Factory.

Business Entity (Business Class)

A Business entity is a domain object that exists independently of other objects within a given business domain. A typical XI-Factory application manages one central (strong) entity, which has sufficient attributes to form a primary key.

Entity Attributes

Every entity is described by a paired set of attributes (attributes, data value -- see *AZUR.MDE.Employees*). For example:

- (Employee.ID, 123)
- (Employee.Name, John Smith)
- (Employee.Email, jsmith@abc.com).

XI-Factory supports all standard data types.

Multi-valued Attributes

Multi-valued attributes can have more than one value. For example, an employee can have one or several hobbies, which must then be normalized either later or directly within the XI-Factory. Multi-valued attributes should be mapped to related attributes (see *AZUR.MDE.Employees-Hobbies*).

Entity Primary Key

A primary key is the field or fields in a table that *uniquely* identifies the record. An everyday example of a primary key is a person's social security number in Canada or the employee email address within a given company.

The power of a relational database system comes from its ability to quickly find and pair information stored in separate tables. In order to do this, each table should include a field or set of fields that uniquely identifies each record stored in the table. This information is called the primary key of the table.

Once you designate a primary key for a table (to ensure uniqueness), the database will prevent any duplicate or Null values from being entered in the primary key fields. There three kinds of primary keys are:

- **AutoNumber:** An AutoNumber field can be set to automatically enter a sequential number as each record is added to the table. Designating such a field as the primary key for a table is the simplest way to create a primary key. For database replication, you must use an AutoNumber.

- **Single-field:** If you have a field that contains unique values (such as ID numbers or email address), you can designate that field as the primary key. If the field you select as primary key does have duplicate or **Null** values, the database won't allow this field as the primary key.

- **Multiple-field:** In situations where you can't guarantee the uniqueness of any single field, you may designate two or more fields as the primary key.

  Note: This type of key is not supported in the XI-Factory analysis phase because of the level of complexity required to manage queries in the application prototypes. You can however easily implement it afterward at the development phase if your DBA enforces it.

The XI-Factory constructor allows you to identify one field as the primary key. The database then will ensure that the field(s) you select are unique and do not contain a null

value (missing or unknown data). If you try to enter a duplicate value, an error is generated.

Specifically, the XI-Factory allows two types of primary keys:

- **Auto-Number:** To ensure the best performance for assigning a primary key in a table is to create an AutoNumber field. However, it can be difficult to remember which numbers apply to which entities (objects).
- **Single-field:** you can use words and numbers to create more meaningful keys, which can make it easier to associate entities with their keys.

Recommendations

When possible, we recommend that you use a single-column surrogate. A surrogate key means that the key's value itself is meaningless with regard to the data. One main reason to do this is that by relating entities via surrogate keys, you never have to worry about the key changing. For example, if you had used an email address and this address changed at some point, then you would need to manually update all the related entities.

Also, you should use numbers (Integer, or GUID) across all entities to simplify operations with the database and the .NET code.

Entity Actions

An action is an operation that represents the performance of a business task in a workflow. An Entity Action is defined as follows:

- It is associated with a transition
- It takes a small amount of time to complete
- It is considered to be non-interruptible

The XI-Factory supports the basic New, Modify, Delete, and Save Actions on any domain entities. You can also define specific Actions with associated functional descriptions (Business rules).

Entity Relationships

The XI-Factory supports the entity relationships between domain entities. From these entity relationships, you can design complete business domain models. For the following examples, we'll assume that the two entities are known as A and B.

Weak Entity

If the existence of entity B depends on the existence of entity A, then B is said to be existence dependent on A (or we say that A is the dominant entity and B is the subordinate entity.).

A weak entity lacks sufficient attributes to form a primary key. Therefore, it lacks the necessary attributes to form *any* key and must *import* attributes from a stronger entity set(s) (see *AZUR.MDE.Employees-Dependant*).

Binary 1:1 Relationships

An entity in A is associated with a single entity in B, and an entity in B is associated with a single entity in A.

For example, an entity in Employees is associated with a single entity in AdditionalInformation, and an entity in AdditionalInformation is associated with a single in Employees (see *AZUR.MDE.Employees-AdditionalInformation*).

Binary 1:N Relationships

An entity in A is associated with any number of entities in B. An entity in B is associated with a single entity in A.

For example, an entity in Employees is associated with any number of entities in Notes. An entity in Notes is associated with a single entity in Employees (see *A.MDE.Employees-Notes*).

Binary N:1 Relationships

An entity in A is associated with a single entity in B. An entity in B is associated with any number of entities in A.

For example, an entity in Employees is associated with a single entity in Departments. An entity in Departments is associated with any number of entities in Employees (see *AZUR.MDE.Empoyee-WorksFor-Departments*).

Binary M:N Relationships

Entities in A and B are associated with any number of entities from within each other.

For example, entities in Employees and Projects are associated with any number of entities from within each other. The appropriate mapping cardinality for a particular relationship set depends on the real world being modeled. To split the M:N relationship, you need to design a new Application (see *AZUR.MDE.EmployeeWorksOnProjects*) to manage both 1:N relationships between Employees and Projects (see below).
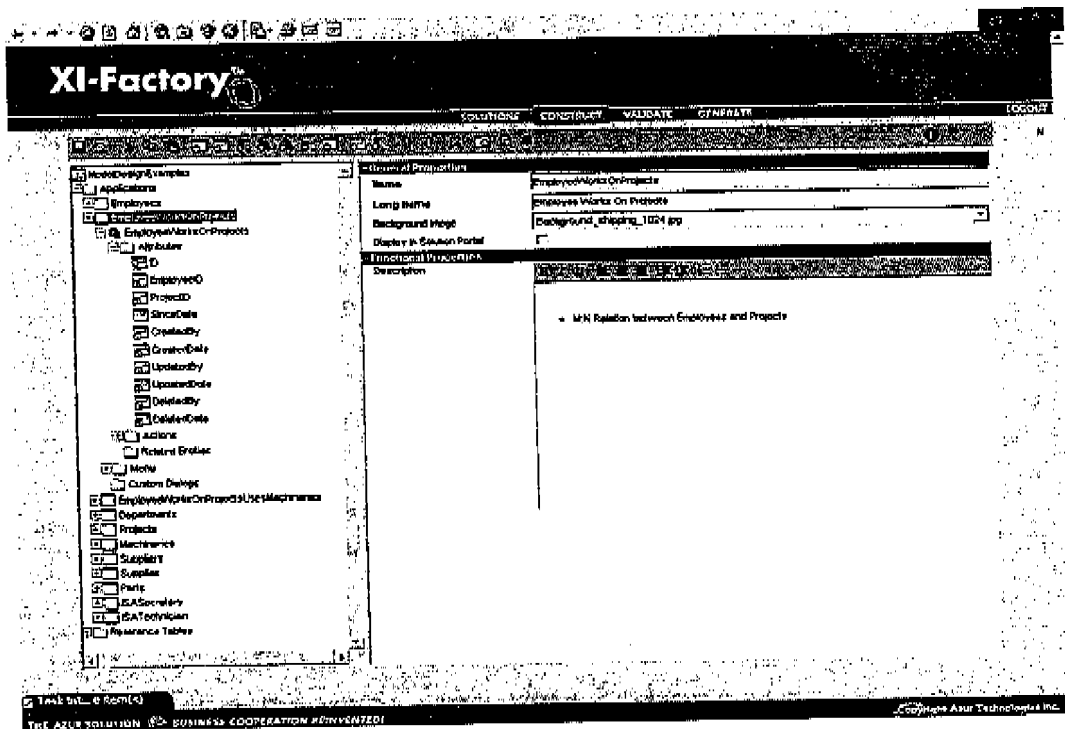
N-ary Relationships

Non-binary relationships can be represented within the XI-Factory. For example, the entities Suppliers, Projects, and Parts can be associated together with the Supplies N-ary relationship (see *AZUR.MDE.Supplies*).

IS A Relationships

This is a sub entity-Super entity relationship which can express the similarities between the entity sets by **generalization**. This is the process of forming containment relationships between a **higher-level** entity set and one or more **lower-level** entity sets.

There are several ways to implement an IS A Relationship. Here are two common examples:

- The entity Employees has an ISA type of attribute (which is a reference table with values like Secretary, Technician, etc.) with optional corresponding attributes (see *AZUR.MDE.Employees*).
- Another example would be if the Employees entity has an ISA association (strong entities with different relational signatures (Secretary, Technician, etc.) with optional corresponding attributes (see *AZUR.MDE.Employees.ISA*).

Multiple Inheritance (MI)

A subclass is inherited from two or more super classes. MI produces *mixin* entities. Although conceptually simple, MI introduces practical complexities. Analysts should use MI with extreme caution to avoid creating an illogical entity. We recommend that because MI is prone to error, you should avoid using it. MI can and should always be transformed into single inheritance with the proper IS A semantic.

[4]: The problem of transforming a multiple inheritance hierarchy into a single inheritance "equivalent" hierarchy can be tackled using the following basic strategies: emancipation, composition, expansion, and *Variant* type, or simulating with a monitor class and flags.

Aggregation

An abstraction through which relationships are treated as higher-level entities. For example, we treat the *EmployeeWorksOnProjects* relationship and the *employee* and *projects* entities as a higher-level entity set. Then we link the *Machineries* entity with the *EmployeeWorksOnProjectsUsesMachineries* relationship (see *AZUR.MDE.EmployeeWorksOnProjectsUsesMachineries*).

Business Model to Object Model Mapping

Once the Business Model is designed and user validated, the XI-Factory helps you automatically generate a stateless N-Tier .NET Solution with corresponding object-oriented projects and underlying domain classes. Within a .NET project, relationships between main entities (Assemblies with Classes) are realized through web services to maximize reuse.

Therefore, from within a standard Software Development Process (Unified Software Development Process (USDP), Rational Unified Process (RUP), etc.), that you can exercise the full UML notation richness to describe and design multifaceted domain behaviors as displayed by the business analyst in the XI-Factory Functional Properties.

Object Model to Relational Model Mapping

Codd first proposed the relational model in 1970 and since then, many software packages rely on relational databases.

The XI-Factory Application database generation must convert a Stateless Object model to a Data Model. Therefore, we must generate well-defined database tables with stored procedures to remain within Relational model constraints. Within the XI-Factory construction phase, the IT business analyst should construct a 3NF normalized business model to optimize the application code and database development phases:

- 1NF (First Normal Form):   no multivalued attributes
- 2NF (Second Normal Form):   no partial dependencies
- 3NF (Third Normal Form):   avoid bad transitive dependencies
- BCNF (Boyce-Codd Normal Form):   strengthens 3NF

See [6] for information on data normalization.

Entities data de-normalization can be achieved using views and web forms when you need optimized user entry forms. The UML [5] and E-R Model [2] mappings to Relational Models are very well defined in software engineering documentation.

XI-Factory™ E-Business Model Design

References

[1] Scott Henninger, Supporting the Domain Lifecycle, Appears in Proceedings of the International Workshop on Computer Aided Software Engineering, Toronto, CA, July 1995, pp. 10-19

[2] Richard Barker, CASE*METHOD, The Entity-Relationship Model, ORACLE.

[3] www.omg.org, OMG-Unified Modeling Language, v1.5

[4] Yania Crespo1, On the Translation of Multiple Inheritance Hierarchies into Single Inheritance Hierarchies, Dept. of Computer Science, Universidad de Valladolid, Spain

[5] Rational White Paper, Mapping Object to Data Models With UML, www.rational.com

[6] Paolo Atzeni, University of Rome, Relational Database Theory, The Benjamin/Cummings Publishing Company, Inc.

[7] Grady Booch, Object-Oriented Analysis and Design, The Benjamin/Cummings Publishing Company, Inc., 1994.

Exhibit C

Achieving Ei and B2Bi, the MDA™ Way!

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

Expanding Upon Software Development Process

JL Lalonde
President, CEO

An AZUR Technologies White paper

Exhibit C

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

Trademarks

The following are trademarks or registered trademarks of their respective companies:

- XI-Factory is a trademark of AZUR Technologies
- XI-Framework is a trademark of AZUR Technologies
- OMG is a trademark of the Object Management Group
- MDA is a trademark of the Object Management Group
- UML is a trademark of the Object Management Group

Exhibit C

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

Table of Contents

Exhibit C

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

Abstract

With the entrenchment of the World Wide Web in modern society, the business world has experienced its own revolution in terms of how they conduct their affairs. Now that online business is becoming increasingly accepted into mainstream society, the focus has now moved from Business-to-Customer (B2C) interactions to higher-level trade partnerships that need to operate in close cooperation to allow their businesses to grow and succeed.

Essentially, **Enterprise Integration (Ei)** and **Business-to-Business Integration** (B2Bi) involve the secured management of business information across a company's internal systems, while synchronizing that information properly with a partner's information systems. Companies, from a broad range of businesses, are taking on B2Bi and recognizing the huge advantages it provides through increase customer and supplier services, lower integration costs, and faster time to market.

Moreover, business strategic alignment or the alignment of information systems strategy with business strategy continues to be ranked as one of the most important issues facing corporations.

But how can we securely deploy dynamic e-business applications rapidly at an affordable and competitive price with total user satisfaction? These applications can include the following:

- e-Order Processing Systems
- e-CRM
- e-Partners Management
- e-Timesheet Management
- e-Incident Management
- e-Room Reservation
- e-Personnel Assignation
- e-Order Scheduling

This White Paper discusses the fundamentals of how to achieve such Business and Business-to-Business integrations economically using web services derived from a Model Driven Architecture™ approach. We present a modern and innovative **Software Fabrication Process** (SFP), with e-collaborative tools, to enhance the regular **Software Development Process** (SDP) both synchronized to quickly realize Ei and B2Bi benefits into your company.

The White Paper illustrates how this processes combination can reduce the software development complexity, time, and cost tenfold, while promoting, as never before, IT and Business Units (BU) people collaboration. As a consequence, this process ensures a successful business strategic alignment.

Exhibit C

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

Since 1995, AZUR Technologies has specialized in automating its Software Fabrication Process for creating and deploying, as quickly as possible, high-quality dynamic e-business applications that meet the business user's expectations.

Achieving Ei and B2Bi, the MDA™ Way

Achieving Ei and B2Bi are easier said than done. Complete enterprise integration is a challenge, especially for global corporations that interact with hundreds to thousands of partners. When faced with so many business partners, the effort to integrate them all is not only expensive and complex, but it is a project that can consume a daunting amount of time. And thanks to an ever-evolving level of technology, the potential for incompatibilities when exchanging digital information makes the task even more complicated.

For small to medium-sized companies, the challenge is also related to the complexity of the software development. Most of the time, these companies do not have the proper software development expertise to adequately integrate their business with their trading partners. When struggling with Ei and B2Bi, all companies (large, medium, small, or new) will invest their time and money to find new solutions, only to find themselves with separate standalone systems that cannot share their resources efficiently.

What they all need is a better way to integrate their systems, while lowering both the level of software development complexity and the overall cost of such essential and multi-faceted system integration.

The Model Driven Architecture™ (see www.omg.org) is a new and stimulating way that can help any size company better integrate its business processes and systems while reducing the software development complexity. In its 2002-2004 forecasts, *PriceWaterhouseCoopers* recognized MDA as highly strategic for companies. Its principle is simple: you need to build your business models in terms of your company objectives rather than the technical environment. This principle consists of creating a Platform Independent Model (PIM) for the company as well as one or more Platform Specific Models (PSM) based on the company's technical environment.

However, to implement an MDA approach in your company, it requires the addition of MDA bridges to automatically transform business models and requirements into dynamic e-business applications, which in turn, promote enterprise integration using web services. All operations must be supervised by an efficient collaborative workflow between the IT and BU personnel.

The Problem Definition

Ei and B2Bi are best-achieved using web services that promote the reuse of heterogeneous resources disseminated inside and outside of the company's networks.

Exhibit C

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

Still, it is very difficult and time consuming to build dynamic e-business applications, consuming web services, with a standard Software Development Process (SDP). Moreover, the cost to develop such high-visibility e-application with a solid foundation and a well defined XInternet N-Tier applicative infrastructure can easily reach beyond the half a million-dollar mark (assuming that everything goes as planned). Currently, dynamic e-business application development, based on web services, is a booming new trend supported by mainly large and medium-sized corporations using a variety of technologies from a variety of vendors.

E-business applications are becoming more and more complex to build now that the Internet is being transformed towards the XInternet ('eXecutable' Internet, see www.forrester.com). Project deadlines are being missed, development budgets are being exceeded, and the costs continue to rise.

Many of the currently available development tools possess a complexity designed for developers, a few for managers, and analysts, but none have been designed specifically for business unit experts. Throughout a software's life cycle, the level of integration for these tools is very low and must be done manually for the most part.

The problem with many of the development products from many IT tool suppliers on the market is that they are difficult to learn and understand, and even more complex to use. Some companies have managed to enhance their solution using Unified Modeling Language (UML) to simplify their tools with some code generation capability.

The general consensus in development circles is that there needs to be a simplification of the object-oriented software development process. IT experts have discussed the complexity the Unified Modeling Language (UML) and have made recommendation to simplify it (for more information, see the 800-page specification (version 2.0) at www.omg.org). However, one of the main conclusions is that you must use a process to guide the modeler in order to use UML effectively.

The level of complexity, coupled with the high cost and time required to build dynamic e-application using the standard Software Development Process, prevents most mid size enterprises from building their own applications successfully. Moreover, the quality and performance of resulting in-house e-application is unpredictable, lacking the proper N-Tier applicative infrastructure needed to guarantee application scalability, robustness, and security.

The AZUR Solution

Realizing that dynamic e-business applications are very complicated to build, we have designed the AZUR Solution to capitalize upon the expertise of a high-caliber team of architects, analysts and developers in a reusable, generic software infrastructure. The AZUR Solution is backed by a Software Fabrication Process (SFP) with productivity tools. SFP speeds up and supports the automated fabrication of e-business applications while reducing both the time and cost required to complete these projects.

Exhibit C

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

SFP hides most of the complexity until it is time to customize the software with the specific structures, business rules, and process integrations. SFP solution allows the business and IT people to be truly and efficiently involved at the center of the analysis phase to ensure strategic alignment.

The SFP is meant to complement the usual Software Development Process (SDP), but you do need to make use of both processes to quickly and efficiently produce e-applications.

- **SFP**: this strategic process automatically generates the dynamic e-biz application generic structures based on business models and requirements.
- **SDP**: this tactical process manually codes the business specific structures and matching business rules.

In order to fabricate an e-application quickly (up to 10 times faster than with SDP alone) while maintaining your high standards of quality, you should make certain that both processes are harmonized.

Software Fabrication Process

Definition

To better understand the SFP; let's define a standard fabrication process:

"A regular manufacturing or fabrication process involves carrying out automatically mass production orders using standard equipments, operations, components, rules, and constraints, while still operating in the most economic manner".

The amount of flexibility you have in this type of automated process depends upon your choice of product configurations.

To mimic a standard fabrication process, we have created (and are still developing) a Software Fabrication Process supported by productivity tools (XI-Factory™ and XI-Framework™) to replicate a regular manufacturing process, but applied to software fabrication. Therefore, the SFP can considerably reduce the time needed to produce e-business applications, while lowering the incurred cost as much as possible.

If automobiles were built like software, we would still be building every single component by hand for every single automobile in the world!

Analysing the Software Fabrication Process

To fully appreciate the Software Fabrication Process, we first need to define the Standard Development Process from which it is based.

Exhibit C

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

An SDP is a set of mostly manual activities that you need to transform a user's requirements into a software system. At AZUR, we adopted the Booch's Notation and Process early on. Since then, we follow the Unified Software Development Process (USDP), which is characterized by being:

- Iterative and incremental
- Use-case driven
- Architecture centered

We also use the Unified Modeling Language™ (UML), the industry-standard language for specifying, visualizing, constructing, and documenting software system artifacts. We have used both USDP and UML approaches to build our own in-house and client-requested software.

The Software Fabrication Process builds upon the regular USDP to offer a more user friendly and automatic process, while promoting a higher level of collaboration between IT and Business Unit people. You can think of the SFP as a process that decreases the complexity of the USDP by providing pre-defined and customizable parameters for inputs, processes, and tools.

The innovative elements that we have added are:

1. The **Model Driven Architecture** approach

2. A web tool to support a **Collaboration Workflow**

3. A **Dynamic e-Business Application Generator**

4. An **N-Tier Applicative Infrastructure**

Generally, the SFP, supported with web-based productivity software tools, enables us to generate 40% to 75% (sometime 100%) of the dynamic e-biz application code directly following the analysis phase. In most cases, after the application is generated, we need to complete the e-application with business rules, reports and system integrations.

When it is time to apply the standard SDP, we use the Unified Software Development Process with its standard inception, elaboration, construction, transition, and evolution phases in an iterative and incremental manner to efficiently complete the application.

The Model Driven Architecture™ Approach

One of the most important inputs of the SFP is the business model with its associated business rules. We introduced the Model Driven Architecture (MDA™) approach into our solution to let business users participate more actively and efficiently to the application model specification, definition, and validation.

Exhibit C

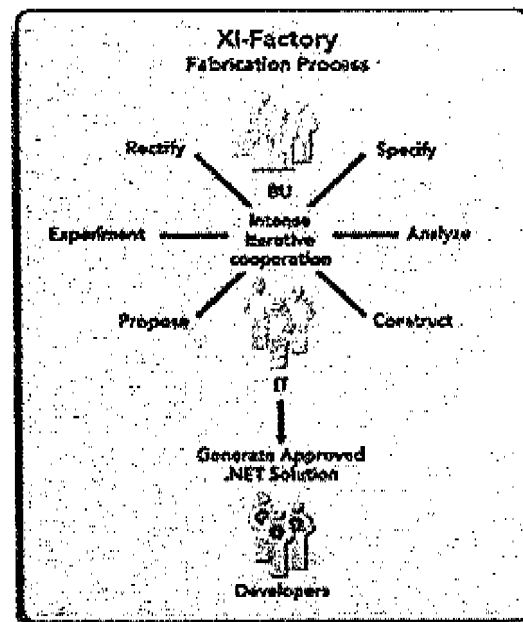A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

With the Software Fabrication Process, we offer an MDA™ bridge that transforms, with a high level of automation, a Platform Independent Model (PIM) into a dynamic e-business application. We can also import a UML model from the IBM Rational Rose Modeler into our solution; the model can then be completed and refined.

The MDA is best realized from a modern collaborative workflow between IT and BU people working together over the web to define and validate business models and requirements, which can, in turn, better define and validate the Ei and B2Bi.

The Collaboration Workflow

To execute the SFP and promote strategic alignment, we have created a collaboration workflow that supports an online software fabrication process and produces a dynamic e-business application in four easy steps:

1. Define the solution.

   - Define solution parameters
   - Import the UML model (optional)
   - Define the security for IT and expert users access rights and roles

2. Construct the solution.

   - Define and refine the business models and rules

3. Preview the solution.

   - Generate automatically the client side of the e-application
   - Experiment the application with an XML database simulation
   - Generate automatically the functional document
   - Provide user feedback into a web analysis log

4. Generate an approved and complete .NET solution.

Our experiences with our clients have shown that following this simple process yields an intense, iterative collaboration or cooperation amongst all the participants in your IT and Business Unit (BU) departments.

After analyzing the business needs, your IT experts can automatically generate an approved .NET solution that developers can finalize and deploy in the approval and production environments. To support the collaborative workflow, we created a strategic web tool for IT teams and their Chief Information Officers (CIO) to reduce IT costs while improving the level of services to the company's BUs.

Exhibit C

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

e-Collaboration Center

Now that we have improved the collaboration workflow between the ITs and BUs, we need to provide the efficient online collaborative functions that can best support this workflow.

What non-technical business users need are adequate tools that help them to validate and precise their business needs. We thus provide an e-Collaboration Center where IT and BU personnel can easily exchange comments and replies (assisted with relevant documents), all of which can be centralized into a single database.

Moreover, they can achieve higher level of collaboration using visual and easy-to-use e-business application prototypes, rather than the complex UML projections (more commonly used by IT teams).

What we've learned from our client projects is that their expert users are not comfortable with a standard Software Development Process promoting the usage of standard UML projections like "Class", "Use Case", or "interaction" diagrams. This is because these projections are either too complex or not user-friendly enough. When working with our clients, most expert users told us that they prefer a **concrete, tangible application** that they can test directly for user validation to any other kind of system projection.

To reflect this business reality, we provide an online mechanism that automatically generates, based on the business models, the e-business application operational prototypes on top of an XML document. This XML document simulates, as much as possible, the application database so that the business users can experiment with the application while it is still in the analysis phase. This process also automatically generates a functional document (HTML and Word formats) with complete business information.

The Dynamic e-Business Application Generator

Once the analysis phase is completed, approved, and the functional document is signed, we can proceed with the generation process. The application generator creates an installation file that the developers can use to deploy the generated .NET solution in their own VS.NET development environment, including:

- User services together with the XInternet web interface
- XML Web services
- Business services
- Data access services
- SQL scripts to generate database tables with generic stored procedures
- A functional document with approved business models and rules

All of these items are supported by an applicative infrastructure and automatically included in your .NET solution. The final result allows you to instantly obtain an

Exhibit C

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

operational .NET solution that can perform basic operations (search, add, modify, delete) on the main and secondary entities (Master-Detail Relationship).

To finish the e-business application, developer must use the functional document information to program the business intelligence and integrate it with the company's (Ei) and partners (B2Bi) heterogeneous business systems.

The Dynamic e-Business Applicative Infrastructure

With its XI-Framework™ migration to .NET technologies, AZUR is offering a modern yet established N-Tier applicative infrastructure able to support variety of clients' e-business applications consuming web services. XI-Framework includes a complete applicative infrastructure composed several major Designs Patterns driven by the *One Web Page Application* concept (see: www.xiazur.com), which has evolved since AZUR's debut to a stable and widely accepted approach in developing efficient dynamic e-business applications.

These structural patterns have been thoroughly validated and tested over a seven-year period and have been endorsed and approved by all of AZUR's clients. Our high-profile and international clients have been able to use this applicative infrastructure to build their systems with predefined structures and patterns.

The XI-Framework is a set of components, system integrations, and documentation that includes the following:

- N-Tier architecture and infrastructure documentation
- Client-side generic structures and components
- Server-side generic system interfaces with methods
- Security management
- Error management
- Workflow management
- Transaction management
- Multilingual management
- E-mail management
- Data Access management
- Reporting management
- Reference data management
- Deployment management

The XI-Framework represents a fundamental and highly strategic achievement for building complex systems from reusable generic structures. This allows for much faster system development, saving the company a great deal of time and money.

Exhibit C

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

To Wrap up on SFP

The AZUR Solution simplifies and accelerates the fabrication of e-business applications and, by using XML Web services, it will facilitate integrating your company's business processes (Ei) with your partner's business processes (B2Bi). The overall Software life cycle and phases are then distributed over these two families of processes and tools. From a management perspective, the SFP and SDP processes both implement the following phases: Inception, Elaboration, Construction, Transition and Evolution.

The SFP produces the first generation of the software, from a highly automatic process. The SDP (the manual process of customization) produces the second generation of the entire software package. Both processes, within their own scope limitations, use a set of standard activities (Specification, Analysis, Design, Implementation, Tests).

The structure of the AZUR-Solution is composed of a Software Fabrication Process supported by of the following tools:

- **Productivity and Collaboration Level**: XI-Factory™
- **N-Tier Applicative Infrastructure and Pattern Level**: XI-Framework™

The structure of the AZUR-Solution requires the following Microsoft technologies:

- **Program Level**: Visual Studio .NET
- **System Level**: Microsoft .NET Framework
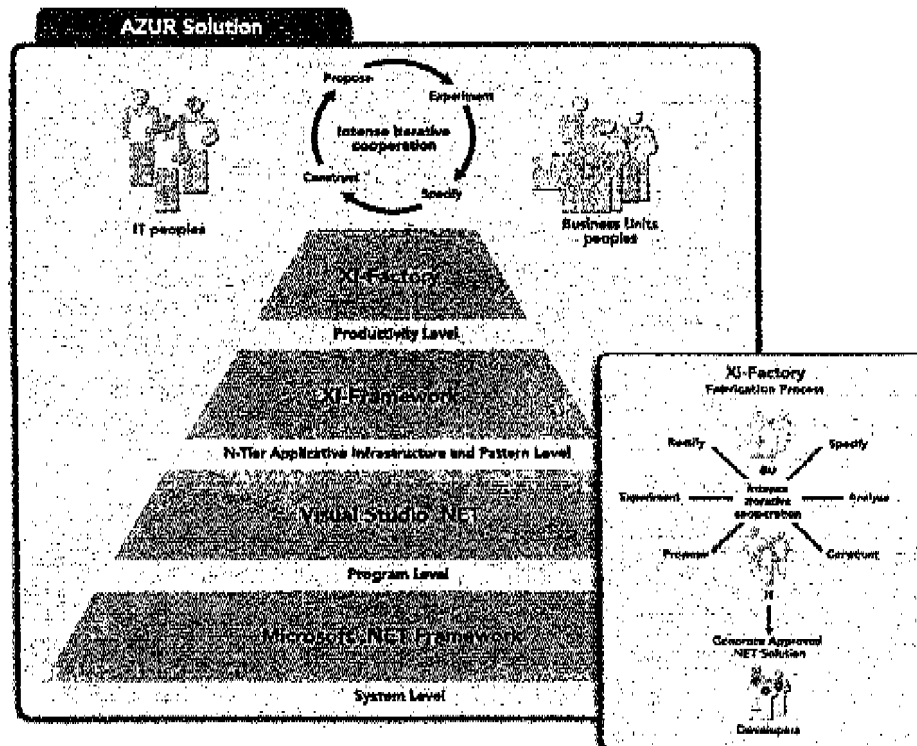
Exhibit C

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

> The AZUR-Solution is therefore an additional layer to Microsoft's program and system levels, aiming at helping IT and Business unit people build less costly and more efficient dynamic e-business applications.

Using SFP: the Numbers Don't Lie

> By choosing the AZUR Solution for a first medium-sized .NET project, your organization can save several hundreds of thousands of dollars. For subsequent .NET projects, your company's efficiency will increase by at least 30%.

> The AZUR Solution is the result of over 10,000 hours of research & development and successful client application deliveries. You can see the return on your investment in a matter of a few weeks or even days!

> The following table shows the return on investment with cost and time savings for a first mid-size project when comparing a typical project against a project augmented with the SFP with tools. These numbers have been carefully evaluated and are backed by years of successful client integration projects.

	Typical Software Development Process (SDP)			Software Fabrication Process (SFP) including: XI-Factory™ and XI-Framework™		
	XIntenet			XIntenet		
Phases	System functions	N-Tier applicative infrastructure	Total	System functions	N-Tier applicative Infrastructure	Total
Specification	10	50	60	2.5	0	2.5
Analysis	30	200	230	20	0	20
Design	30	200	230	25	0	25
Implementation	60	600	660	35	0	35
Tests	30	200	230	25	0	25
Days	160	1250	1410	107.5	0	107.5
Costs (USD) $450/day	$72,000	$562,500[1]	$634,500	$48,375	0	$48,375

Project ROI calculation for a first mid-size application:
• Project total cost with typical SDP : $634,500
• Project total cost with SFP combined with SDP: $48,375
• Savings of: 92.4%

Project savings for any subsequent application:
• ($48,375 / $72,000) *100 ~ 67 % for savings of ~ 33 % per additional application.

[1] *Conservative estimate based on a seasoned and experienced IT team for these types of projects.*

Exhibit C

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

Conclusion

This White paper has presented the fundamentals of how to achieve Enterprise Integration and Business-to-Business integration economically through web services derived form a Model Driven Architecture™ approach. We have presented a modern and innovative Software Fabrication Process, with e-collaborative tools, to enhance the regular Software Development Process both synchronized to quickly realize Ei and B2Bi benefits into your company. We also showed how this processes combination can reduce tenfold the software development complexity, time and cost while promoting, as never before, IT and Business Units (BU) people collaboration and as a consequence, ensuring the business strategic alignment.

The Software Fabrication Process with e-Collaborative Tools puts an emphasis on automating the creation, validation and generation of high-quality dynamic e-business applications that meet the business user's expectations. Usually, the SFP, supported with the AZUR XI-Factory™ and XI-Framework™ software tools, enables us to generate 40% to 75% (sometime 100%) of the dynamic e-biz application code directly following the analysis phase. The resulting benefits for your company are significant and can vary from:

- Finally, Ei and B2Bi are **truly achievable**, at a fraction of the cost, from a mature yet adaptable solution based of widely accepted web services.

- **Very fast time** to market dynamic e-business application deployment fully aligned with business needs.

- **Optimization and harmonization of relations** between the IT and business unit departments through an efficient collaboration workflow supported by a unique e-Collaboration Center.

- **Save a considerable amount of time and money** since the N-Tier applicative infrastructure becomes available at a small fraction of the usual development cost and effort.

- **Quick and efficient** initiation of .NET operational projects very useful for learning how to write Visual Basic or C# code for the .NET platform.

You are invited to try the SFP with the AZUR XI-Factory™ and XI-Framework™ tools at the AZUR **TRY IT CENTER**. You can also experiments several Dynamic e-Businesses Sample Applications to better evaluate the XInternet sensation and performance.

In another White paper, we will present a concrete demonstration of SFP and how it can convert secluded client-server Applications into XInternet e-Business Applications in a very short period of time. This conversion demonstration will be applied to the very well-known Microsoft Access Northwind application transforming it into the **XI-Northwind** e-Business Application.

Exhibit C

A Software Fabrication Process with E-Collaborative Tools to Promote Strategic Alignment

References

Gunjan Samtani and Dimple Sadhwani, B2Bi and Web Services, An Intimidating Task?,Web Services Architect, January 2 2002

Paul P. Tallong, Boston College USA, Kenneth L Kraemer, University of California USA, Investigating the Relationship between Strategic Alignment and IT Business Value: The Discovery of a Paradox. Idea Group Publishing, 2003.

Richard D. Buchanan, Aligning Enterprise Architecture and IT Investments with Corporate Goals, META Group.

David Frankel, Using Model-Driven Architecture™ to Develop Web Sevices, IONA Technologies, April 2002.

Philippe Kruchten, A Rational Development Process, Crosstalk, 9 (7) 1996, pp.11-16.

Gary Police, Using the Rational Unified Process for small Projects: Expanding upon eXtreme Programming, A Rational White paper.

Steven Witkop, Driving business agility with Model Driven Architecture, EDS.

Ramkumar Ramaswamy, Variability in software development process families, Software Quality Professional, March 2001